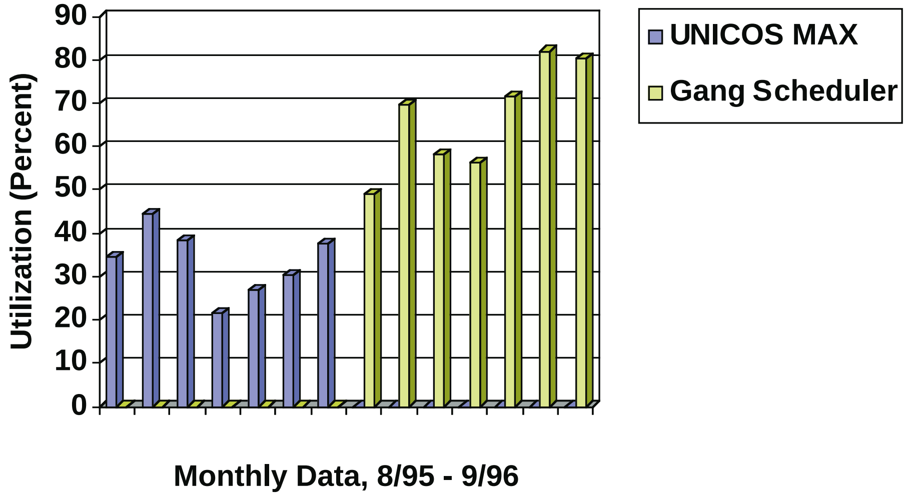
University of California
Lawrence Livermore
National Laboratory

Gang Scheduler

Timesharing on a Massively Parallel Supercomputer



This chart shows the dramatic improvement in throughput of the LLNL Cray T3D. Larger jobs are executed and throughput has dramatically increased while providing very good interactivity. Utilization reported is the percentage of all CPU cycles available which are delivered to customer codes. Weekly utilization rates have reached over 90 percent.

Technology

The Gang Scheduler, under development at the Lawrence Livermore National Laboratory, supports timesharing of the parallel machine resources provided by a CRAY T3D massively parallel supercomputer.

Without appropriate scheduling software, processors are allocated in a space-sharing mode, and lockout can occur frequently. The Gang Scheduler combines preemptive processor scheduling with the ability to move jobs within a pool of available processors to achieve both timely response for interactive computing and long run times for production computing.

Most scalable parallel systems available today support space sharing of system resources among contending jobs, but do not support timesharing the entire parallel machine. Cray Research Inc.'s (CRI's) CRAY T3D is no exception, since its default mode of operation is the allocation of job partitions from the pool of available processors. These job partitions are held until the job relinquishes them, effectively locking out any other use for those processors. With such a processor allocation mechanism, the computational requirements of long-running production jobs directly conflict with those of interactive development jobs. Only a preemptive processor scheduler could satisfy the requirements of all clients.

The Gang Scheduler is a preemptive processor scheduler. It schedules processors and barrier circuits for all jobs, which are classified by access requirements as follows:

- Interactive class jobs receive the most responsive service.
- Debug class jobs are not preempted and receive responsive service.
- Production class jobs receive better throughput but less responsive service.
- Benchmark class jobs are not preempted but may experience the least responsive service.
- Standby class jobs are allocated processors that would otherwise be unused.

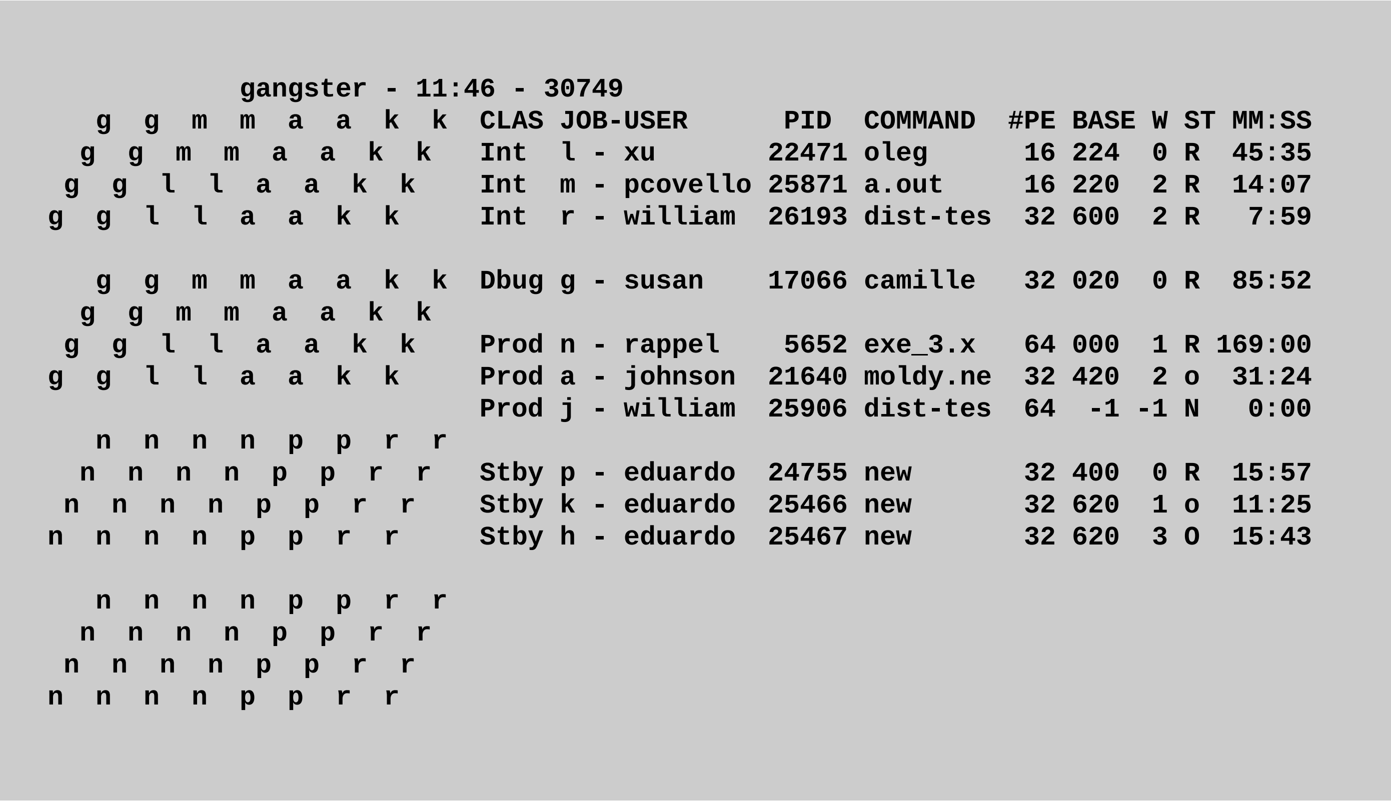
Jobs may be preempted to satisfy the requirements for responsive service and fair distribution of resources among these jobs. Preempted jobs will have their state moved to disk and will relinquish their processors. Other jobs will then be allocated processors from the pool of those available.

Features

Thrashing is eliminated by jobs being guaranteed their assigned processes for some period of time after being granted access. This *do-not-disturb* time is a function of the number of processors assigned and the job class. It ensures that the time necessary to save and restore a job's state is small in comparison to its execution time. Once a job's *do-not-disturb* time has been exhausted, any other job waiting to be loaded having an equal or higher job class can be scheduled to replace it. Jobs within each job class are sorted in order of time waiting to be loaded. The longest-waiting jobs within each job class are given the highest priority for loading.

Each job class has a configurable maximum wait time to ensure sufficient interactivity. After a job has waited to be loaded for the maximum wait time, a block of processors will be reserved for it. Jobs occupying the reserved processors will be preempted as soon as their *do-not-disturb* time has been exhausted. When the last of these processors has been made available, the longest-waiting job will be allocated the reserved processors.

Other configuration options control the maximum number of processors that may be assigned to a single job, the maximum number of processors available for each job class, and the maximum number of processors available for large jobs (jobs requiring large numbers of processing ele-



```
            gangster - 11:46 - 30749
   g  g  m  m  a  a  k  k  CLAS JOB-USER      PID  COMMAND  #PE BASE W ST MM:SS
  g  g  m  m  a  a  k  k   Int  l - xu       22471 oleg      16 224  0 R  45:35
 g  g  l  l  a  a  k  k    Int  m - pcovello 25871 a.out     16 220  2 R  14:07
g  g  l  l  a  a  k  k     Int  r - william  26193 dist-tes  32 600  2 R   7:59

   g  g  m  m  a  a  k  k  Dbug g - susan    17066 camille   32 020  0 R  85:52
  g  g  m  m  a  a  k  k
 g  g  l  l  a  a  k  k    Prod n - rappel    5652 exe_3.x   64 000  1 R 169:00
g  g  l  l  a  a  k  k     Prod a - johnson  21640 moldy.ne  32 420  2 o  31:24
                           Prod j - william  25906 dist-tes  64  -1 -1 N   0:00
   n  n  n  n  p  p  r  r
  n  n  n  n  p  p  r  r   Stby p - eduardo  24755 new       32 400  0 R  15:57
 n  n  n  n  p  p  r  r    Stby k - eduardo  25466 new       32 620  1 o  11:25
n  n  n  n  p  p  r  r     Stby h - eduardo  25467 new       32 620  3 O  15:43

   n  n  n  n  p  p  r  r
  n  n  n  n  p  p  r  r
 n  n  n  n  p  p  r  r
n  n  n  n  p  p  r  r
```

Sample GANGSTER display identifying jobs in the system and assigned processors. The node map is on the left. A dot or letter denotes each node (two processing elements on the T3D): a dot indicates the node is not in use, a letter designates the job currently occupying that node. On the right is a summary of all jobs. The ST field reports the job‘s state: R = running; o = being moved out to disk; O = on disk; i = being moved into processors; N = new job, not assigned processors.

ments). These configuration options may be altered in real-time, permitting different behavior at different times of the day or on different days.

Client Interface

The execution of a user‘s job is passed through a Gang Scheduler interface. This interface is built upon the CRI default interface and is upwardly compatible with it. The interface registers the job with the Gang Scheduler and waits for an assignment of processors and barrier circuit before continuing. This typically takes a matter of seconds for interactive or debug class jobs.

GANGSTER Tool

We provide users with an interactive tool, GANGSTER, for observing the state of the system and controlling some aspects of their jobs. GANGSTER communicates with the Gang Scheduler to determine the state of the machine‘s processors and individual jobs. GANGSTER‘s three-dimensional node map displays the status of each node (each node consists of two processing elements on the T3D). GANGSTER‘s job summary reports the state of each job, including jobs moving between processors and disk (see the figure). Users can use GANGSTER to change the class of their own jobs or to explicitly move their jobs into processors or out to disk.

Availability

The Gang Scheduler is a collaborative effort between Cray Research, Inc., and Lawrence Livermore National Laboratory. It is available for use only on CRAY T3D computers.

For additional information, contact
Moe Jette, 510-423-4856, jette@llnl.gov.

Work performed under the auspices of the U.S. DOE by LLNL under contract No. W-7405-ENG-48